In [1]:
```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
ifood_df = pd.read_csv('ifood.csv')
ifood_df
```

Out[1]:

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProduct
0	58138.0	0	0	58	635	88	54
1	46344.0	1	1	38	11	1	
2	71613.0	0	0	26	426	49	12
3	26646.0	1	0	26	11	4	2
4	58293.0	1	0	94	173	43	11
...	...	...	...	...	...	...	.
2200	61223.0	0	1	46	709	43	18
2201	64014.0	2	1	56	406	0	3
2202	56981.0	0	0	91	908	48	21
2203	69245.0	0	1	8	428	30	21
2204	52869.0	1	1	40	84	3	6

2205 rows × 39 columns

In [2]:
```
# Some Intial Cleaning
ifood_df.sort_values(by='Income', ascending=False)
columns_to_drop = ["education_2n Cycle", "education_Basic", "education_Gradu
]
ifood_df.drop(columns=columns_to_drop, axis=1, inplace=True)
ifood_df
```

Out[2]:

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProduct
0	58138.0	0	0	58	635	88	54
1	46344.0	1	1	38	11	1	
2	71613.0	0	0	26	426	49	12
3	26646.0	1	0	26	11	4	2
4	58293.0	1	0	94	173	43	11
...	...	...	...	...	...	...	.
2200	61223.0	0	1	46	709	43	18
2201	64014.0	2	1	56	406	0	3
2202	56981.0	0	0	91	908	48	21
2203	69245.0	0	1	8	428	30	21
2204	52869.0	1	1	40	84	3	6

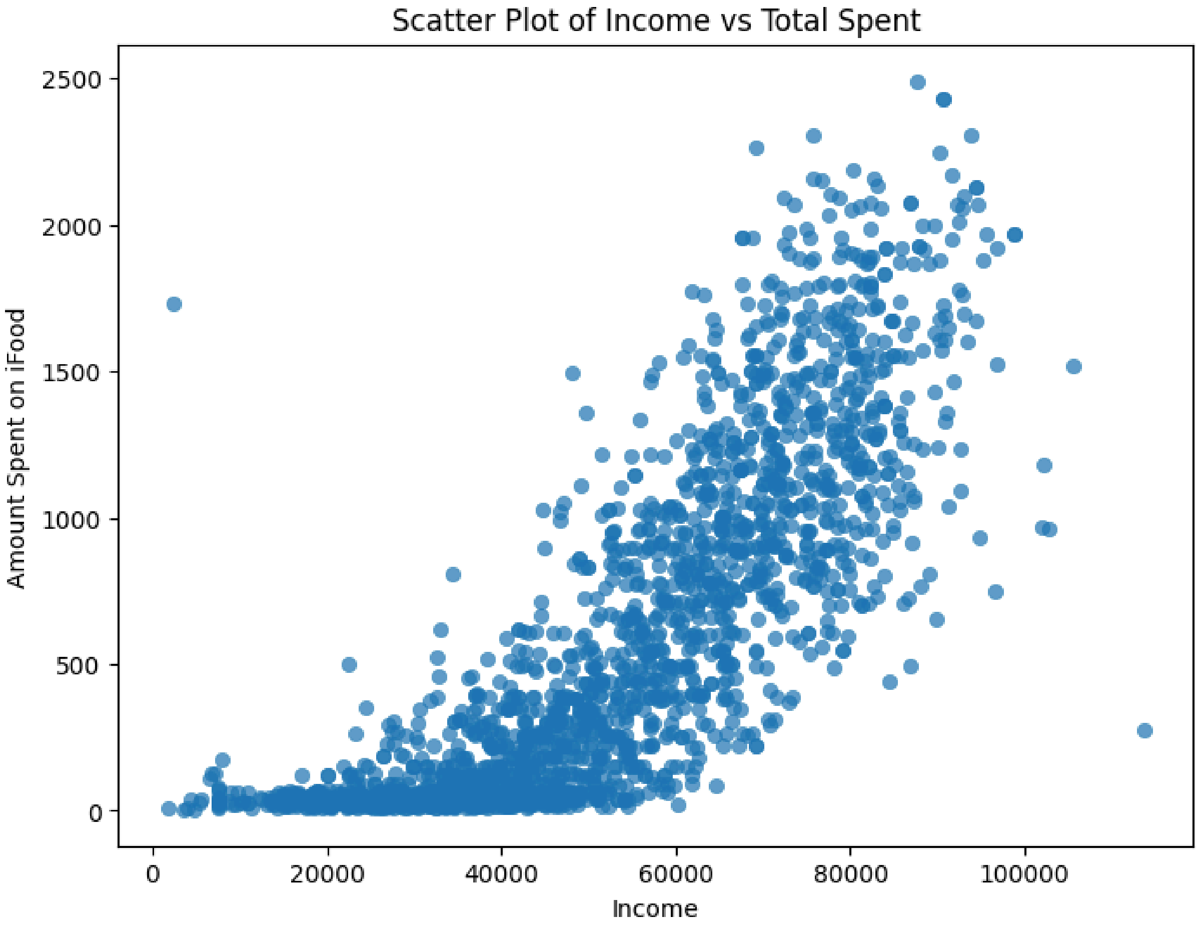
2205 rows × 34 columns

In [3]:

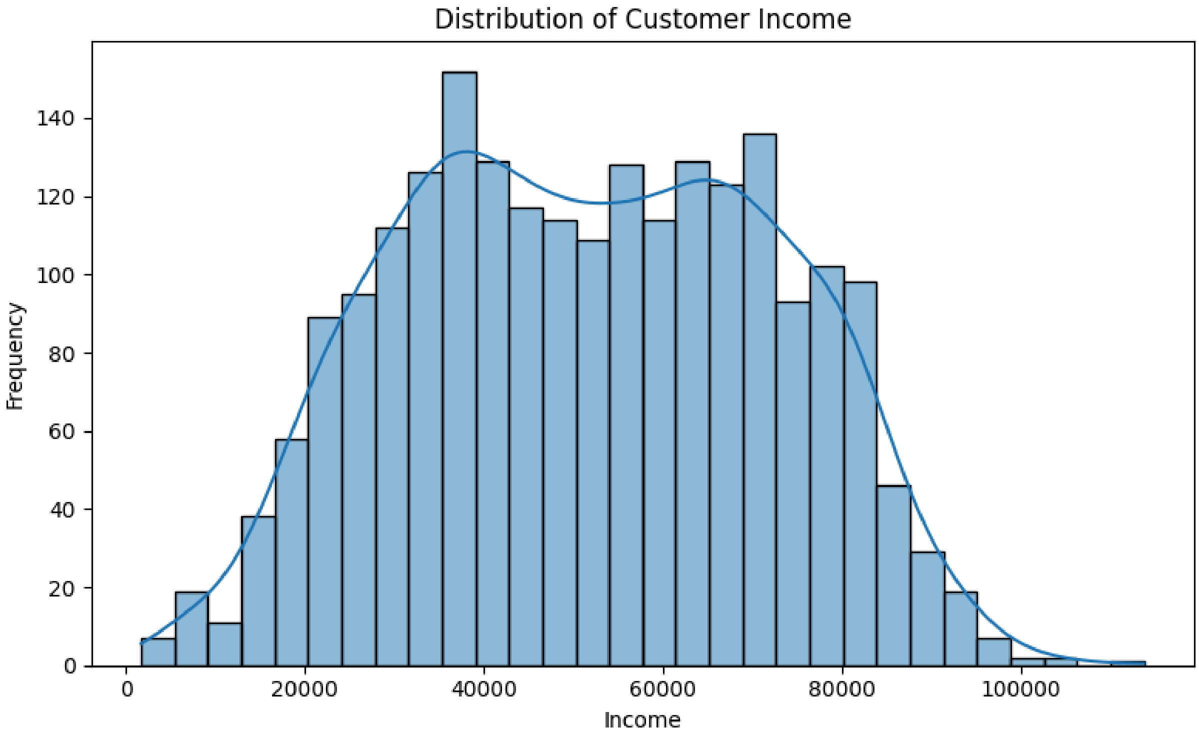
```
x_col = 'Income'
y_col = 'MntTotal'
plt.figure(figsize=(8, 6))
sns.scatterplot(data=ifood_df, x='Income', y='MntTotal', alpha=0.7, edgecolo
plt.title(f"Scatter Plot of {'Income'} vs {'Total Spent'}")
plt.xlabel('Income')
plt.ylabel('Amount Spent on iFood')
```

Out[3]:

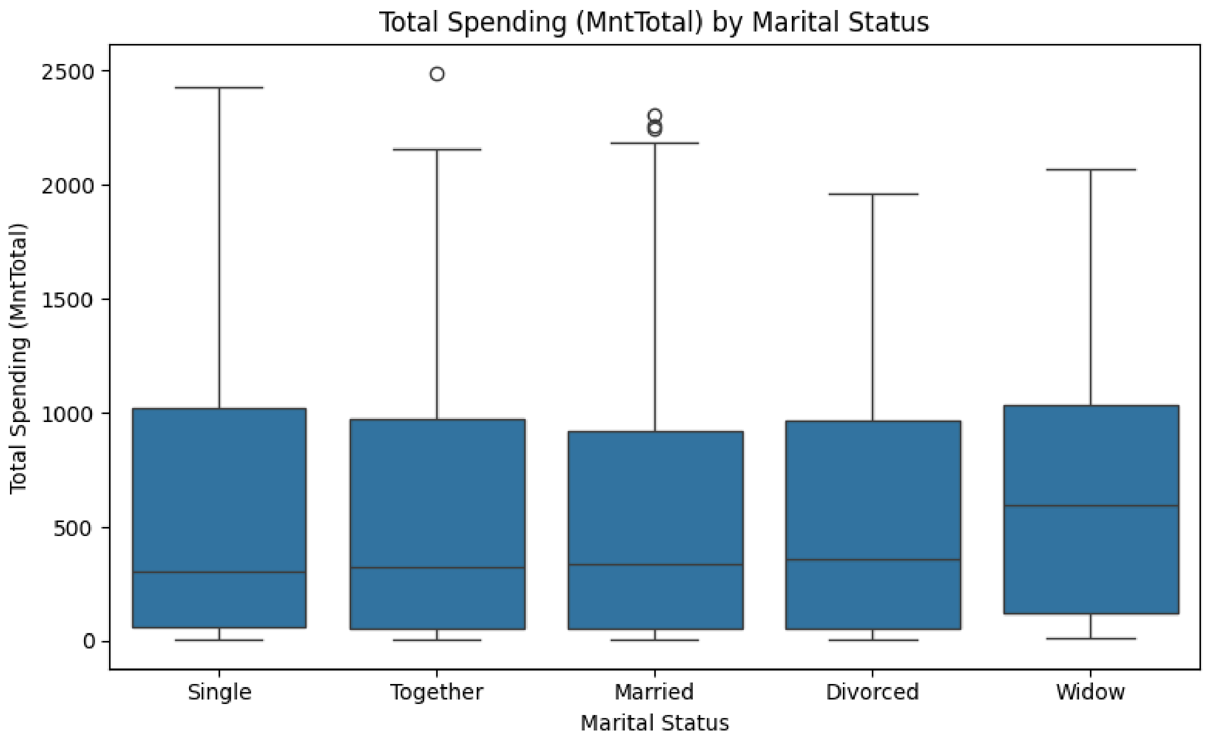
```
Text(0, 0.5, 'Amount Spent on iFood')
```



In [4]:
```
plt.figure(figsize=(8, 5))
sns.histplot(ifood_df["Income"], bins=30, kde=True)
plt.title("Distribution of Customer Income")
plt.xlabel("Income")
plt.ylabel("Frequency")
plt.tight_layout()
plt.show()
```



```
In [5]: marital_status_df = ifood_df.copy()
        # Fix naming to make more clear for Visual
        marital_names = {
            "marital_Divorced": "Divorced",
            "marital_Married": "Married",
            "marital_Single": "Single",
            "marital_Together": "Together",
            "marital_Widow": "Widow"}
        # Create a categorical column for marital status
        marital_status_df["Marital_Status"] = ( marital_status_df[list(marital_names
        # boxplot
        plt.figure(figsize=(8, 5))
        sns.boxplot(data=marital_status_df, x="Marital_Status", y="MntTotal")
        plt.title("Total Spending (MntTotal) by Marital Status")
        plt.xlabel("Marital Status")
        plt.ylabel("Total Spending (MntTotal)")
        plt.tight_layout()
        plt.show()
```



In [ ]:

```
# Model Starts Below
```

In [10]:

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, confusion_matrix, classification
import matplotlib.pyplot as plt
import seaborn as sns

# Load the iFood dataset
ifood_df = pd.read_csv('ifood.csv')

#check:
ifood_df.head()
```

Out[10]:

	Income	Kidhome	Teenhome	Recency	MntWines	MntFruits	MntMeatProducts
0	58138.0	0	0	58	635	88	546
1	46344.0	1	1	38	11	1	6
2	71613.0	0	0	26	426	49	127
3	26646.0	1	0	26	11	4	20
4	58293.0	1	0	94	173	43	118

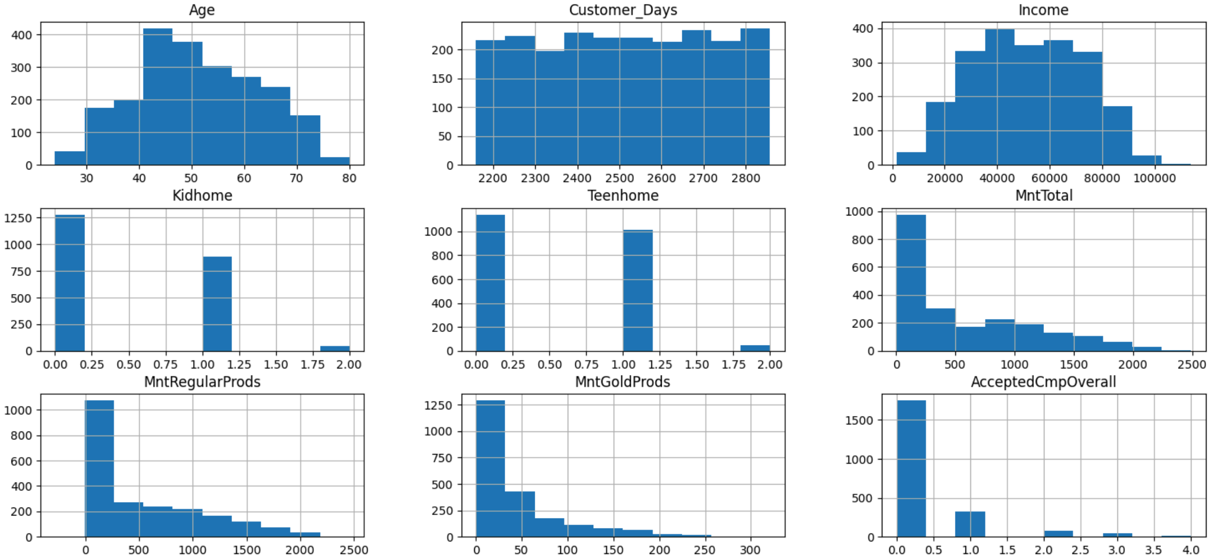
5 rows × 39 columns

In [7]:

```
ifood_df.hist(column = ['Age', 'Customer_Days', 'Income', 'Kidhome', 'Teenho
```

Out[7]:

```
array([[<Axes: title={'center': 'Age'}>,
        <Axes: title={'center': 'Customer_Days'}>,
        <Axes: title={'center': 'Income'}>],
       [<Axes: title={'center': 'Kidhome'}>,
        <Axes: title={'center': 'Teenhome'}>,
        <Axes: title={'center': 'MntTotal'}>],
       [<Axes: title={'center': 'MntRegularProds'}>,
        <Axes: title={'center': 'MntGoldProds'}>,
        <Axes: title={'center': 'AcceptedCmpOverall'}>]], dtype=object)
```



In [12]:

```
#1. Explore how spending features affect purchase (customer response)
spending_features = [
    'MntFishProducts', 'MntMeatProducts', 'MntFruits',
    'MntSweetProducts', 'MntWines', 'MntGoldProds']

X = ifood_df[spending_features + ['MntTotal']]
y = ifood_df['Response']

spending_df = ifood_df[spending_features + ['MntTotal', 'Response']]
```

In [14]:

```
from sklearn.model_selection import train_test_split

train_spend, test_spend = train_test_split(spending_df, test_size=0.2, rando
```

In [15]:

```
#Split train and test
y_train_spend = train_spend['Response']
X_train_spend = train_spend.drop(columns=['Response'])

y_test_spend = test_spend['Response']
X_test_spend = test_spend.drop(columns=['Response'])
```

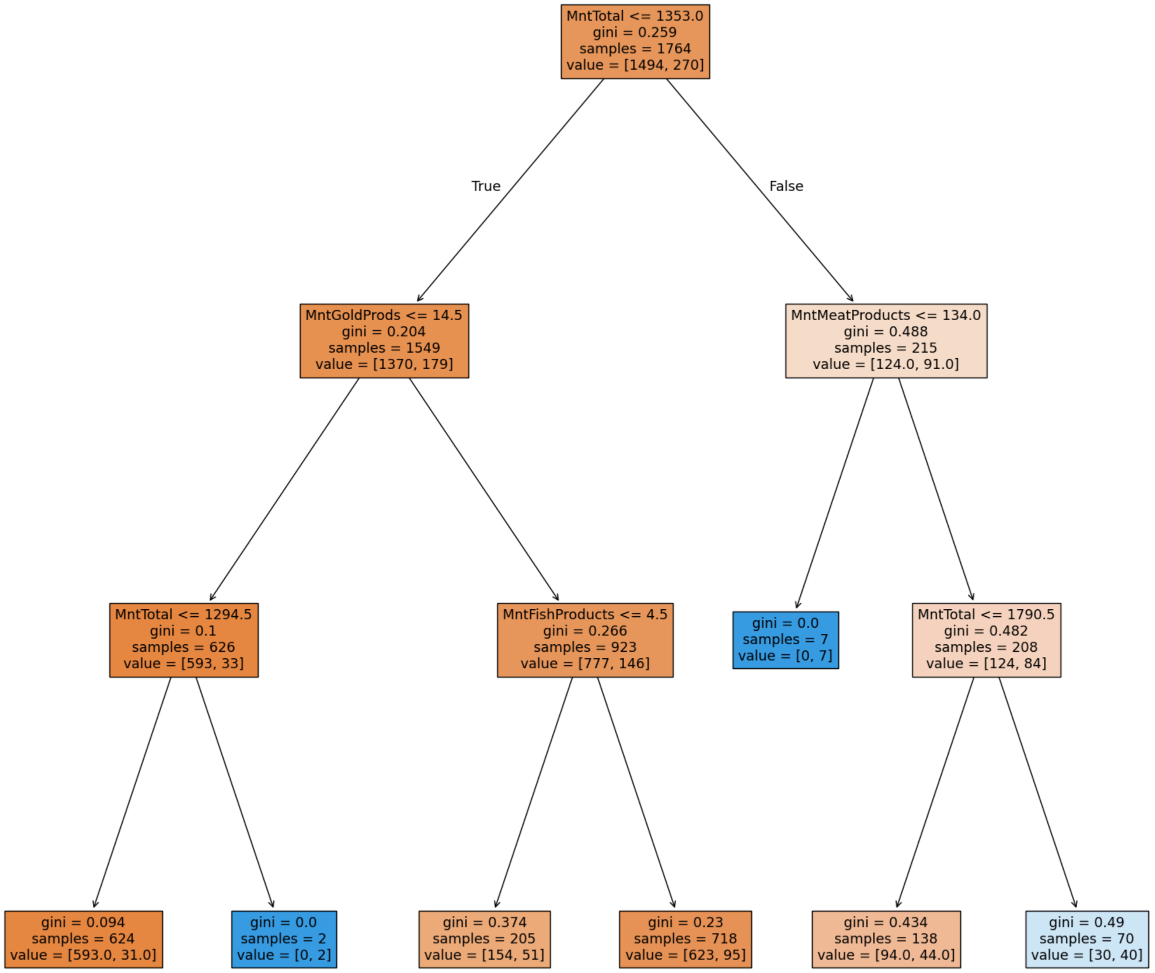
In [16]:

```
# Train Decision Tree
from sklearn.tree import DecisionTreeClassifier, plot_tree

T = DecisionTreeClassifier(max_depth=3, random_state=42)
T.fit(X_train_spend, y_train_spend)

train_score = T.score(X_train_spend, y_train_spend)
test_score = T.score(X_test_spend, y_test_spend)
```

In [17]:

```
print('Score on train:', train_score)
print('Score on test:', test_score)

fig, ax = plt.subplots(1, figsize = (20, 20))
p = plot_tree(T, filled = True, feature_names = X_train_spend.columns)
```

```
Score on train: 0.8577097505668935
Score on test: 0.854875283446712
```



In [19]:
```
#2. Explore how different channels affect purchase (customer response)
channel_features = [
    'NumDealsPurchases', 'NumWebPurchases',
    'NumCatalogPurchases', 'NumStorePurchases']

channel_df = ifood_df[channel_features + ['Response']]
```

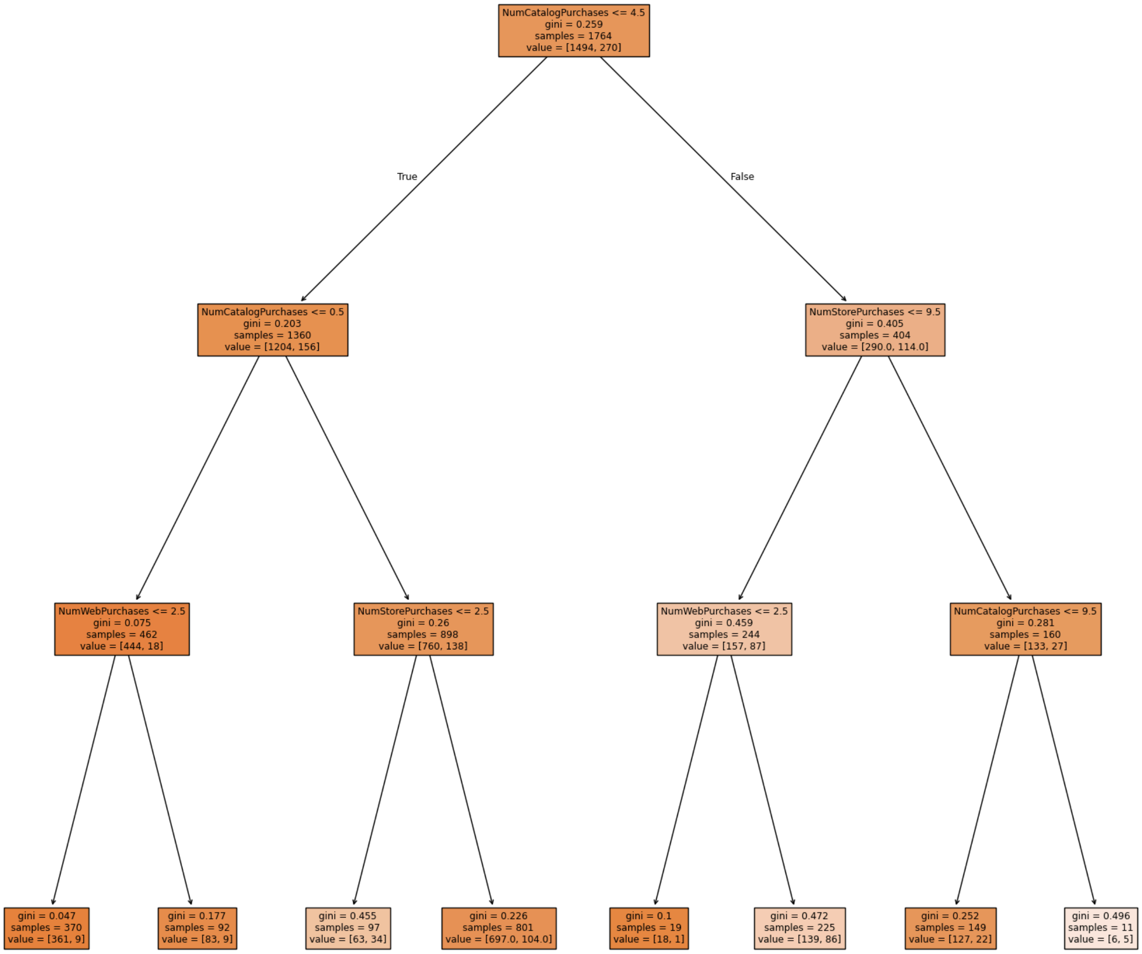
In [20]:
```
#Split train and test
train_channel, test_channel = train_test_split(channel_df, test_size=0.2, ra

y_train_channel = train_channel['Response']
X_train_channel = train_channel.drop(columns=['Response'])

y_test_channel = test_channel['Response']
X_test_channel = test_channel.drop(columns=['Response'])
```

In [21]:
```
# Train Decision Tree
from sklearn.tree import DecisionTreeClassifier, plot_tree
```

```
T = DecisionTreeClassifier(max_depth=3, random_state=42)
T.fit(X_train_channel, y_train_channel)

train_score = T.score(X_train_channel, y_train_channel)
test_score = T.score(X_test_channel, y_test_channel)
```

```
In [22]: print('Score on train:', train_score)
         print('Score on test:', test_score)

         fig, ax = plt.subplots(1, figsize = (20, 20))
         p = plot_tree(T, filled = True, feature_names = X_train_channel.columns)
```

```
Score on train: 0.8469387755102041
Score on test: 0.8571428571428571
```



```
In [ ]:
```